BHAVANA CK

1BM20CS403

CSE-4A

PROGRAM 1

INSURANCE DATABASE

Consider the Insurance database given below. The data types are specified.

PERSON (driver_id: String, name: String, address: String)
CAR (reg_num: String, model: String, year: int)
ACCIDENT (report_num: int, accident_date: date, location: String)
OWNS (driver_id: String, reg_num: String)
PARTICIPATED (driver_id: String,reg_num: String, report_num: int, damage_amount: int)

i) Create the above tables by properly specifying the primary keys and the foreign keys.
ii)Enter at least five tuples for each relation.
iii)Demonstrate how you
    a.Update the damage amount to 25000 for the car with a specific reg-num(example 'K A053408') for which the accident report number was 12.
    b.Add a new accident to the database.
iv)Find the total number of people who owned cars that involved in accidents in 2008.
v)Find the number of accidents in which cars belonging to a specific model (example)were involved.

```
CREATE DATABASE Insurance;

USE Insurance;
```

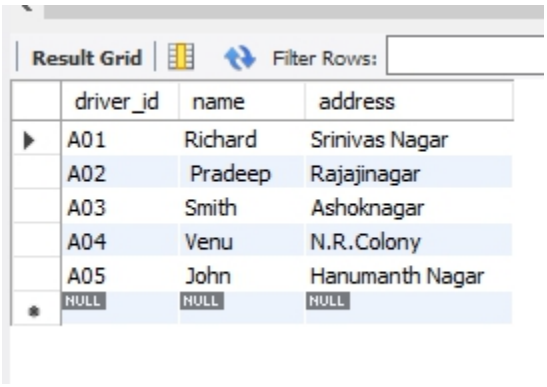
------ Create the above tables by properly specifying the primary keys and the foreign keys.

```
CREATE TABLE PERSON
(
DRIVER_ID VARCHAR(15),
NAME VARCHAR(30) NOT NULL,
ADDRESS VARCHAR(30),
PRIMARY KEY(DRIVER_ID)
);

CREATE TABLE CAR(
REG_NUM VARCHAR(15),
MODEL VARCHAR(15),
YEAR INT,
PRIMARY KEY (REG_NUM)
);

CREATE TABLE ACCIDENT(
REPORT_NUM INT,
ACCIDENT_DATE DATE,
LOCATION VARCHAR(30),
PRIMARY KEY (REPORT_NUM)
);

CREATE TABLE OWNS(
DRIVER_ID VARCHAR(15),
```

```
REG_NUM VARCHAR(15),
FOREIGN KEY (DRIVER_ID) REFERENCES PERSON(DRIVER_ID),
FOREIGN KEY (REG_NUM) REFERENCES CAR(REG_NUM)
);

CREATE TABLE PARTICIPATED(
DRIVER_ID VARCHAR(15),
REG_NUM VARCHAR(15),
REPORT_NUM INT,
DAMAGE_AMOUNT INT,
FOREIGN KEY (DRIVER_ID) REFERENCES PERSON(DRIVER_ID),
FOREIGN KEY (REG_NUM) REFERENCES CAR(REG_NUM),
FOREIGN KEY (REPORT_NUM) REFERENCES ACCIDENT(REPORT_NUM)
);
```

------ Enter at least five tuples for each relation.

```
INSERT INTO PERSON
VALUES("A01","Richard","Srinivas Nagar"),("A02","Pradeep","Rajaji Nagar"),
("A03","Smith","Ashok Nagar"),("A04","Venu","N R Colony"),
("A05","Jhon","Hanumanth Nagar");
SELECT * FROM PERSON;
```



driver_id	name	address
A01	Richard	Srinivas Nagar
A02	Pradeep	Rajajinagar
A03	Smith	Ashoknagar
A04	Venu	N.R.Colony
A05	John	Hanumanth Nagar
NULL	NULL	NULL

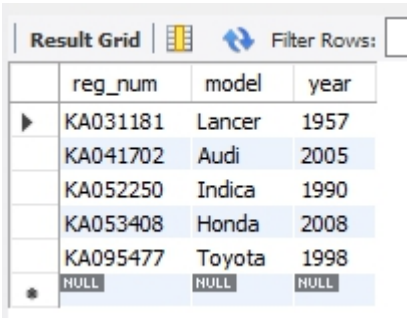
INSERT INTO CAR

VALUES("KA052250","Indica",1990),("KA031181","Lancer",1957),

("KA095477","Toyota",1998),("KA053408","Honda",2008),

("KA041702","Audi",2005);

SELECT * FROM CAR;



reg_num	model	year
KA031181	Lancer	1957
KA041702	Audi	2005
KA052250	Indica	1990
KA053408	Honda	2008
KA095477	Toyota	1998
NULL	NULL	NULL

INSERT INTO OWNS

VALUES("A01","KA052250"),("A02","KA053408"),

("A03","KA031181"),("A04","KA095477"),

("A05","KA041702");

SELECT * FROM OWNS;

Result Grid | Filter Rows:

driver_id	reg_num
A04	KA031181
A05	KA041702
A01	KA052250
A02	KA053408
A03	KA095477
NULL	NULL

INSERT INTO ACCIDENT

VALUES(11,'2003-01-01',"Mysore Road"),(12,'2004-02-02',"South end Circle"),

(13,'2003-01-21',"Bull Temple Road"),(14,'2008-02-17',"Mysore Road"),

(15,'2005-03-04',"Kanakpura Road");

SELECT * FROM ACCIDENT;

Result Grid | Filter Rows:

report_no	accident_date	location
11	2003-01-01	Mysore Road
12	2004-02-02	Southend Circle
13	2003-01-21	Bulltemple Road
14	2008-02-17	Mysore Road
15	2005-03-04	Kanakpura Road
NULL	NULL	NULL

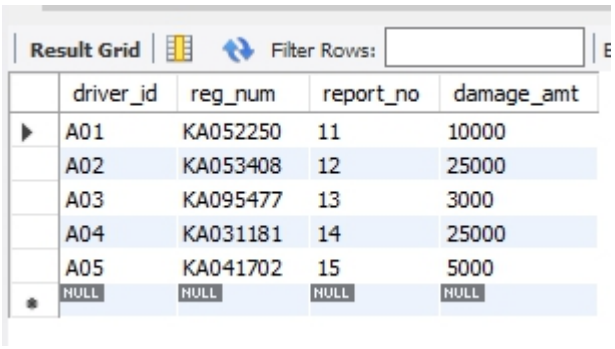
INSERT INTO PARTICIPATED

VALUES("A01","KA052250",11,10000),("A02","KA053408",12,50000),

("A03","KA031181",13,25000),("A04","KA095477",14,3000),

("A05","KA041702",15,5000);

SELECT * FROM PARTICIPATED;

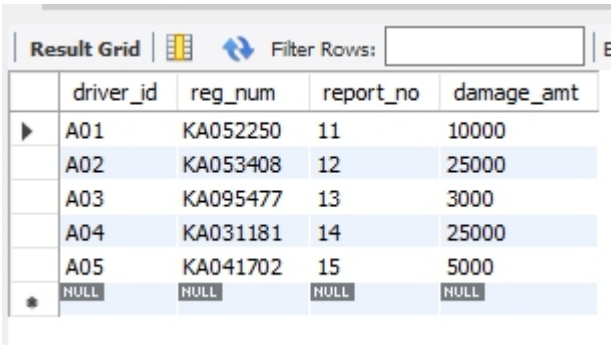


Result Grid | Filter Rows:

driver_id	reg_num	report_no	damage_amt
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	3000
A04	KA031181	14	25000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

------ Update the damage amount to 25000 for the car with a specific reg-num(example 'K A053408') for which the accident report number was 12.

```
UPDATE PARTICIPATED

SET DAMAGE_AMOUNT=25000

WHERE REG_NUM="KA053408";
```

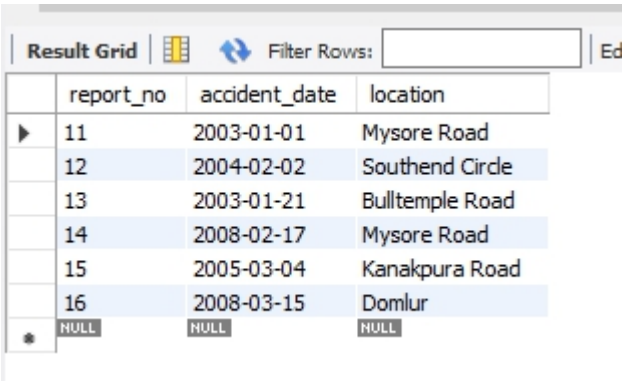


Result Grid | Filter Rows:

driver_id	reg_num	report_no	damage_amt
A01	KA052250	11	10000
A02	KA053408	12	25000
A03	KA095477	13	3000
A04	KA031181	14	25000
A05	KA041702	15	5000
NULL	NULL	NULL	NULL

------ Add a new accident to the database.

```
INSERT INTO ACCIDENT

VALUES (16,"2009-05-12","Belagavi");

SELECT * FROM ACCIDENT;
```

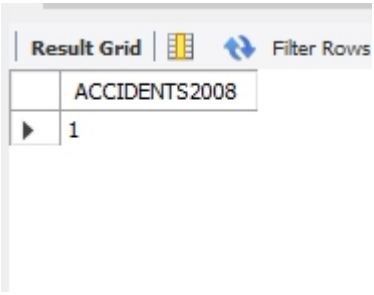


Result Grid | Filter Rows: | Ed

report_no	accident_date	location
11	2003-01-01	Mysore Road
12	2004-02-02	Southend Circle
13	2003-01-21	Bulltemple Road
14	2008-02-17	Mysore Road
15	2005-03-04	Kanakpura Road
16	2008-03-15	Domlur
NULL	NULL	NULL

------ Find the total number of people who owned cars that involved in accidents in 2008.

```
SELECT COUNT(ACCIDENT_DATE) AS ACCIDENTS2008 FROM ACCIDENT

WHERE YEAR(ACCIDENT_DATE)=2008;
```

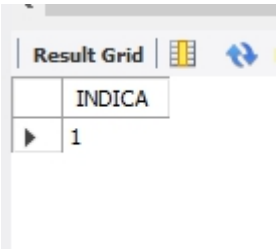


Result Grid | Filter Rows

ACCIDENTS2008
1

------- Find the number of accidents in which cars belonging to a specific model (example)were involved

```
SELECT COUNT(MODEL) AS INDICA FROM car

WHERE MODEL="Indica";
```



Result Grid |

INDICA
1

PROGRAM 2

BANKING ENTERPRISE DATABASE

Consider the following database for a banking enterprise.

Branch (branch-name: String, branch-city: String, assets: real)
BankAccount(accno: int, branch-name: String, balance: real)
BankCustomer (customer-name: String, customer-street: String, customer-city: String)
Depositer(customer-name: String, accno: int)
Loan (loan-number: int, branch-name: String, amount: real)

i. Create the above tables by properly specifying the primary keys and the foreign keys.
ii. Enter at least five tuples for each relation.
iii. Find all the customers who have at least two accounts at the *Main* branch (ex. SBI_ResidencyRoad).
iv. Find all the customers who have an account at *all* the branches located in a specific city (Ex. Delhi).
v. Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

```
create database sample11;

use sample11;


CREATE TABLE branch

(

branch_name VARCHAR(20),

branch_city VARCHAR(20),

assets REAL,

PRIMARY KEY(branch_name)

);


CREATE TABLE accounts

(

acc_no INT,

branch_name VARCHAR(50),

balance REAL,
```

```
PRIMARY KEY(acc_no),
FOREIGN KEY(branch_name) REFERENCES branch(branch_name)
ON UPDATE CASCADE ON DELETE CASCADE
);

CREATE TABLE customer
(
 customer_name VARCHAR(20),
customer_street VARCHAR(50),
customer_city VARCHAR(20),
PRIMARY KEY(customer_name)
);

CREATE TABLE depositor
(
 customer_name VARCHAR(20),
acc_no INT,
PRIMARY KEY(customer_name, acc_no),
FOREIGN KEY(customer_name) REFERENCES customer(customer_name)
ON UPDATE CASCADE ON DELETE CASCADE,
FOREIGN KEY(acc_no) REFERENCES accounts(acc_no)
ON UPDATE CASCADE ON DELETE CASCADE
);

CREATE TABLE loan
(
 loan_number INT,
```

branch_name VARCHAR(50),

amount REAL,

PRIMARY KEY(loan_number),

FOREIGN KEY(branch_name) REFERENCES branch(branch_name)

ON UPDATE CASCADE ON DELETE CASCADE

);

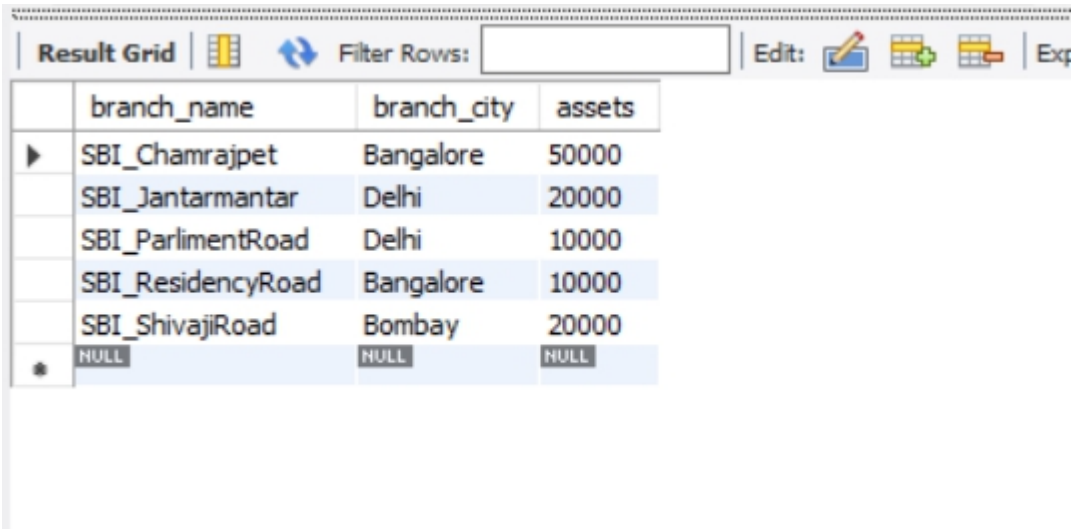
INSERT INTO branch

VALUES ('SBI_Chamrajpet','Bangalore',50000),('SBI_ResidencyRoad','Bangalore',10000),

('SBI_ShivajiRoad','Bombay',20000),('SBI_ParlimentRoad','Delhi',10000),

('SBI_Jantarmantar','Delhi',20000);

SELECT * FROM branch;



branch_name	branch_city	assets
SBI_Chamrajpet	Bangalore	50000
SBI_Jantarmantar	Delhi	20000
SBI_ParlimentRoad	Delhi	10000
SBI_ResidencyRoad	Bangalore	10000
SBI_ShivajiRoad	Bombay	20000
NULL	NULL	NULL

INSERT INTO accounts

VALUES (1,'SBI_Chamrajpet',2000),(2,'SBI_ResidencyRoad',5000),

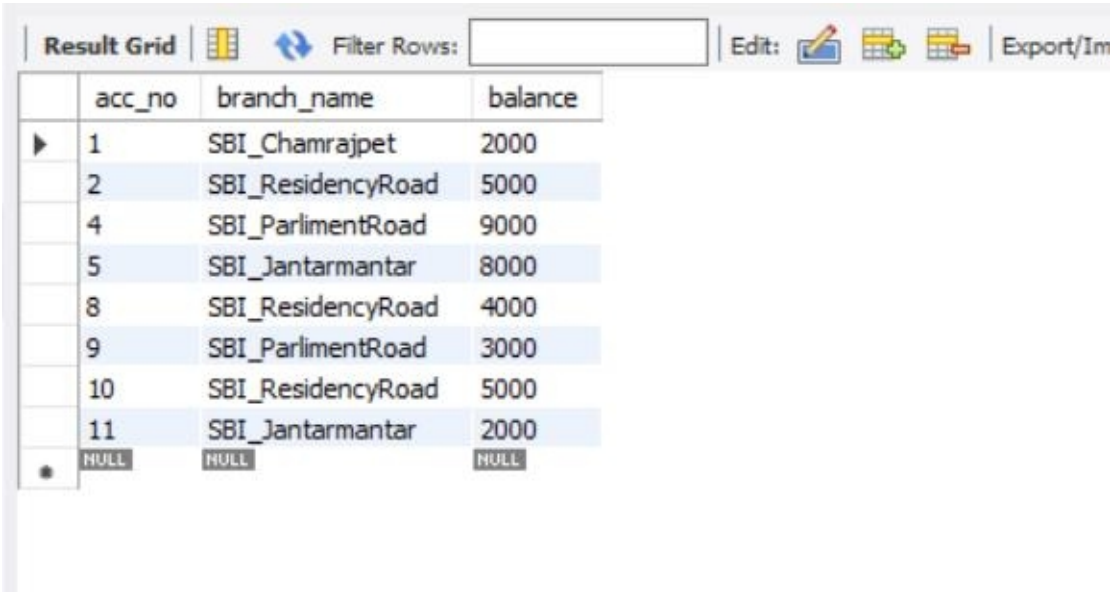
(3,'SBI_ShivajiRoad',6000),(4,'SBI_ParlimentRoad',9000),

(5,'SBI_Jantarmantar',8000),(6,'SBI_ShivajiRoad',4000),

(8,'SBI_ResidencyRoad',4000),(9,'SBI_ParlimentRoad',3000),

(10,'SBI_ResidencyRoad',5000),(11,'SBI_Jantarmantar',2000);

SELECT * FROM accounts;



acc_no	branch_name	balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
NULL	NULL	NULL

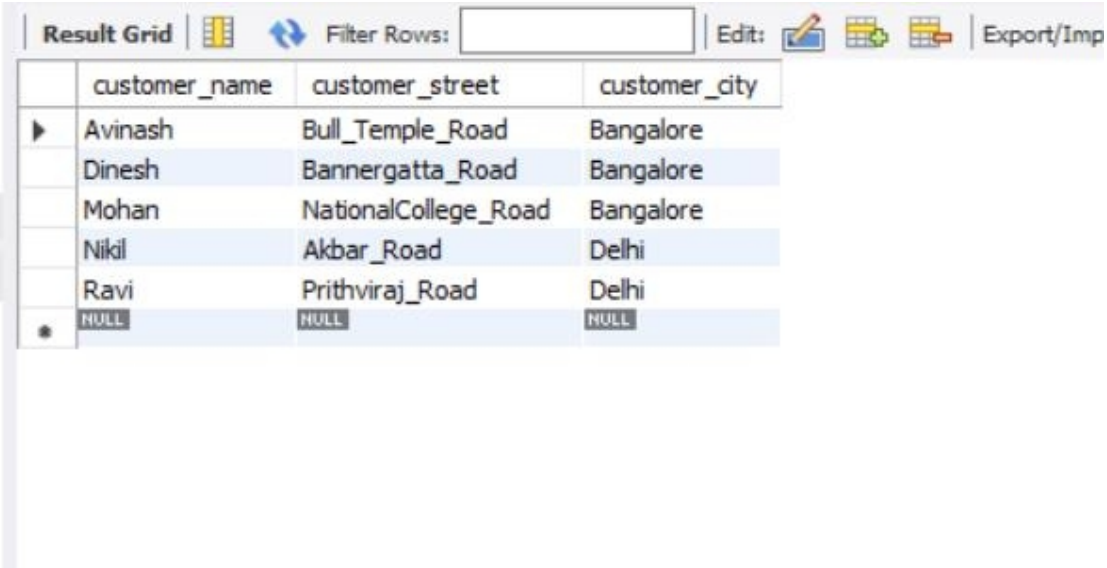
INSERT INTO customer

VALUES('Avinash','Bull_Temple_Road','Bangalore'),('Dinesh','Bannergatta_Road','Bangalore'),

('Mohan','NationalCollege_Road','Bangalore'),('Nikil','Akbar_Road','Delhi'),

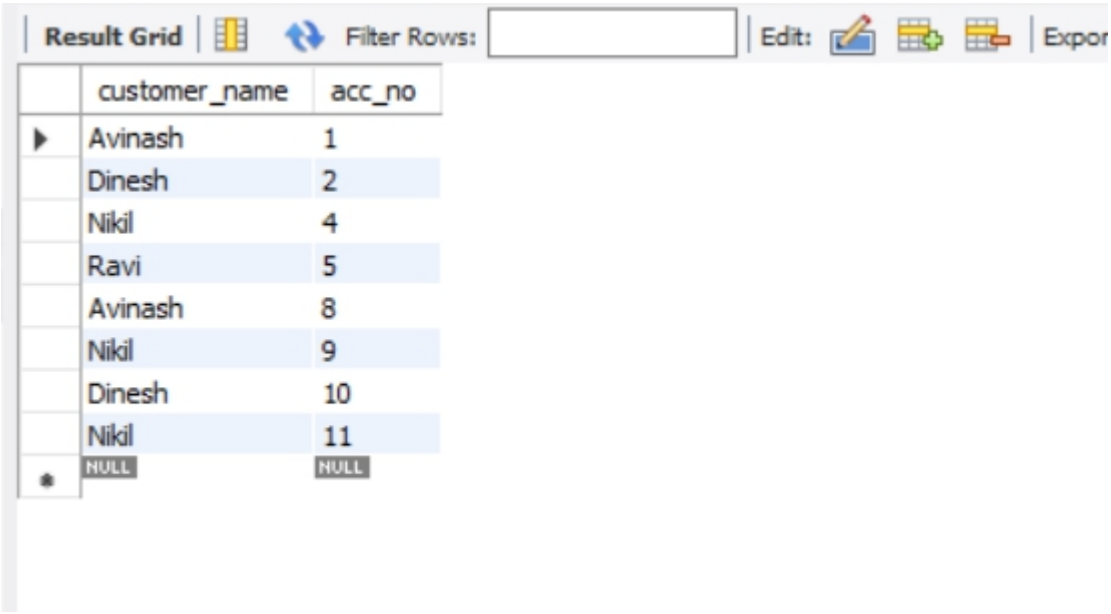
('Ravi','Prithviraj_Road','Delhi');

SELECT * FROM customer;



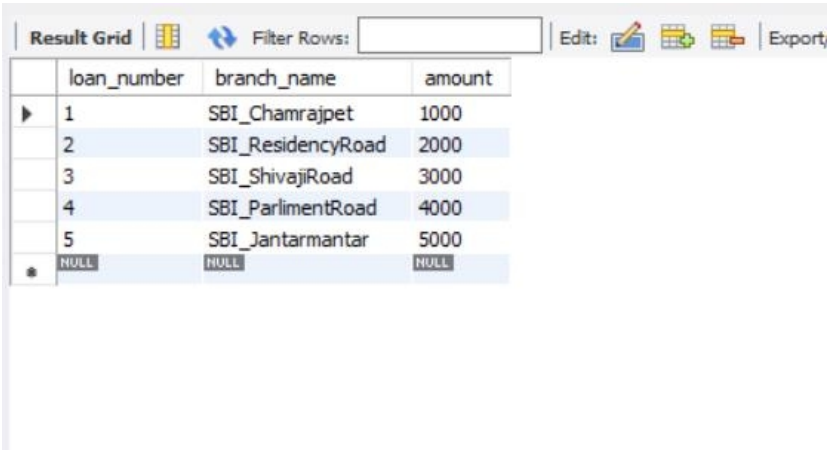
customer_name	customer_street	customer_city
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Mohan	NationalCollege_Road	Bangalore
Nikil	Akbar_Road	Delhi
Ravi	Prithviraj_Road	Delhi
NULL	NULL	NULL

```
INSERT  INTO  depositor

VALUES ('Avinash',1),('Dinesh',2),

('Nikil',4),('Ravi',5),

('Avinash',8),('Nikil',9),

('Dinesh',10),('Nikil',11);


SELECT * FROM depositor;
```



customer_name	acc_no
Avinash	1
Dinesh	2
Nikil	4
Ravi	5
Avinash	8
Nikil	9
Dinesh	10
Nikil	11
NULL	NULL

```
INSERT INTO loan

 VALUES (1,'SBI_Chamrajpet',1000),(2,'SBI_ResidencyRoad',2000),

(3,'SBI_ShivajiRoad',3000),(4,'SBI_ParlimentRoad',4000),

(5,'SBI_Jantarmantar',5000);


SELECT * FROM loan;
```

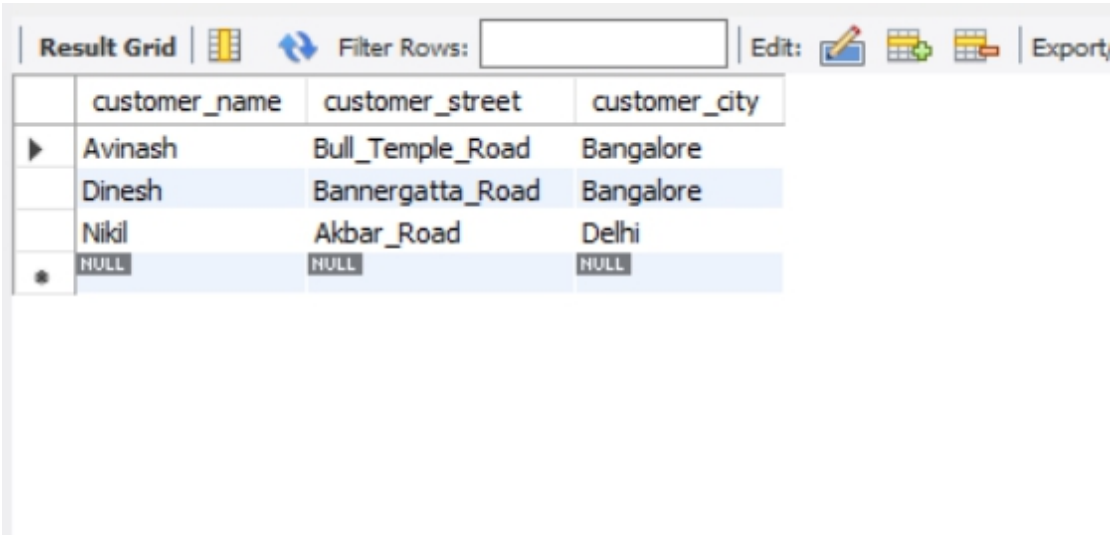


loan_number	branch_name	amount
1	SBI_Chamrajpet	1000
2	SBI_ResidencyRoad	2000
3	SBI_ShivajiRoad	3000
4	SBI_ParlimentRoad	4000
5	SBI_Jantarmantar	5000
NULL	NULL	NULL

.

------- Find all the customers who have at least two accounts at the *Main* branch (ex. SBI_ResidencyRoad).

SELECT * FROM customer WHERE customer_name IN(SELECT customer_name FROM depositor group by customer_name having COUNT(customer_name)>=2);



customer_name	customer_street	customer_city
Avinash	Bull_Temple_Road	Bangalore
Dinesh	Bannergatta_Road	Bangalore
Nikil	Akbar_Road	Delhi
NULL	NULL	NULL

------Find all the customers who have an account at *all* the branches located in a specific city (Ex. Delhi).

SELECT d.customer_name FROM accounts a, depositor d,branch b WHERE d.acc_no=a.acc_no AND b.branch_name=a.branch_name AND b.branch_city="Delhi" GROUP BY d.customer_name having count(distinct b.branch_name)=(SELECT COUNT(branch_name) FROM branch WHERE branch_city="Delhi");

Result Grid | Filter Rows: | Export

customer_name
Nikil

------- Demonstrate how you delete all account tuples at every branch located in a specific city (Ex. Bombay).

```
DELETE FROM ACCOUNTS WHERE branch_name IN(SELECT branch_name FROM BRANCH
WHERE branch_city='Bombay');
```

Result Grid | Filter Rows: | Edit: | Export/Im

acc_no	branch_name	balance
1	SBI_Chamrajpet	2000
2	SBI_ResidencyRoad	5000
4	SBI_ParlimentRoad	9000
5	SBI_Jantarmantar	8000
8	SBI_ResidencyRoad	4000
9	SBI_ParlimentRoad	3000
10	SBI_ResidencyRoad	5000
11	SBI_Jantarmantar	2000
NULL	NULL	NULL

PROGRAM 3

SUPPLIER DATABASE

Consider the following schema:

SUPPLIERS(sid: integer, sname: string, address: string)

PARTS(pid: integer, pname: string, color: string)

CATALOG(sid: integer, pid: integer, cost: real)

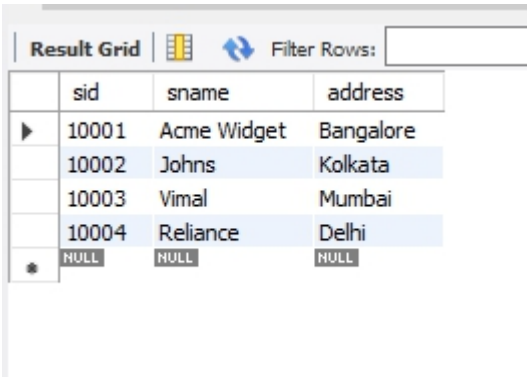
The Catalog relation lists the prices charged for parts by Suppliers.

Write the following queries in SQL:

i. Find the pnames of parts for which there is some supplier.
ii. Find the snames of suppliers who supply every part.
iii. Find the snames of suppliers who supply every red part.
iv. Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.
v. Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).
vi. For each part, find the sname of the supplier who charges the most for that part.

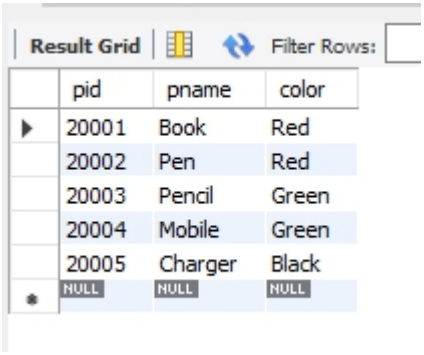
```
create database supplier;

use supplier;


create table suppliers(

sid int not null,

sname varchar(20) not null,

address varchar(20) not null,

primary key(sid)

);


create table parts(

pid int not null,

pname varchar(20) not null,
```

```
color varchar(10) not null,
primary key(pid)
);

create table catalog(
sid int not null,
pid int not null,
cost real not null,
primary key(sid,pid),
foreign key(sid)references suppliers(sid),
foreign key(pid)references parts(pid)
);

insert into suppliers
values(10001,"Acme Widget","Bangalore"),(10002,"Johns","Kolkata"),
(10003,"Vimal","Mumbai"),(10004,"Reliance","Delhi");
```



sid	sname	address
10001	Acme Widget	Bangalore
10002	Johns	Kolkata
10003	Vimal	Mumbai
10004	Reliance	Delhi
NULL	NULL	NULL

```
insert into parts
values (20001,"Book","Red"),(20002,"Pen","Red"),
(20003,"Pencil","Green"),(20004,"Mobile","Green"),
(20005,"Charger","Black");
```



Result Grid | Filter Rows:

pid	pname	color
20001	Book	Red
20002	Pen	Red
20003	Pencil	Green
20004	Mobile	Green
20005	Charger	Black
NULL	NULL	NULL

insert into catalog

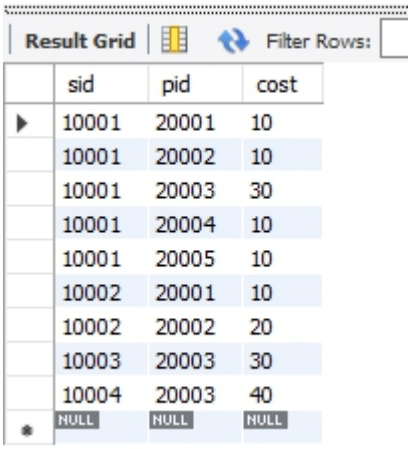
values(10001,20001,10),(10001,20002,10),

(10001,20003,30),(10001,20004,10),

(10001,20005,10),(10002,20001,10),

(10002,20002,20),(10003,20003,30),

(10004,20003,40);



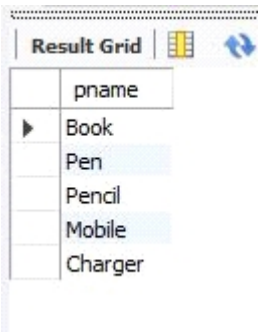
Result Grid | Filter Rows:

sid	pid	cost
10001	20001	10
10001	20002	10
10001	20003	30
10001	20004	10
10001	20005	10
10002	20001	10
10002	20002	20
10003	20003	30
10004	20003	40
NULL	NULL	NULL

-----Find the pnames of parts for which there is some supplier.

select distinct pname from parts,catalog

where catalog.pid= parts.pid and sid is not null;



pname
Book
Pen
Pencil
Mobile
Charger

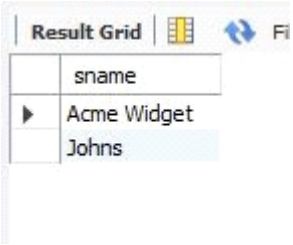
-----Find the snames of suppliers who supply every part.

```
select suppliers.sname,catalog.sid from suppliers,catalog

where suppliers.sid = catalog.sid

group by suppliers.sname

having count(catalog.sid)=(select count(pid) from parts);
```

sname	sid
Acme Widget	10001

----Find the snames of suppliers who supply every red part.

```
select distinct s.sname from suppliers s,catalog c

where s.sid=c.sid and c.pid in(select pid from parts where color ="Red");
```

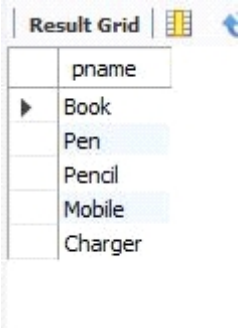


sname
Acme Widget
Johns

-----Find the pnames of parts supplied by Acme Widget Suppliers and by no one else.

```
select pname from parts ,catalog

where parts.pid=catalog.pid and sid in (select sid from suppliers where sname ="Acme
```

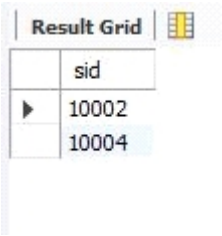
Widget");



pname
Book
Pen
Pencil
Mobile
Charger

-----Find the sids of suppliers who charge more for some part than the average cost of that part (averaged over all the suppliers who supply that part).

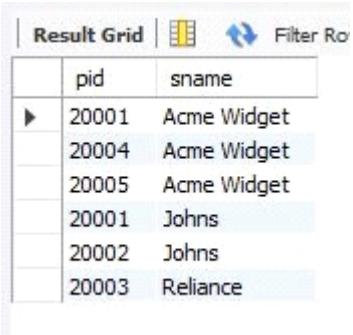
```
select c.sid from catalog c

where  c.cost>(select avg(cost)from catalog where pid = c.pid);
```



sid
10002
10004

----For each part, find the sname of the supplier who charges the most for that part.

```
select c.pid,s.sname from suppliers s,catalog c

where s.sid =c.sid and c.cost =(select max(cost) from catalog where pid =c.pid);
```



pid	sname
20001	Acme Widget
20004	Acme Widget
20005	Acme Widget
20001	Johns
20002	Johns
20003	Reliance

PROGRAM 4

STUDENT FACULTY DATABASE

Consider the following database for student enrollment for course :

STUDENT(snum : integer, sname : string, major: string, lvl : string, age: integer)

CLASS(cname: string, meets at: time, room: string, fid: integer)

ENROLLED(snum: integer, cname: string)

FACULTY(fid: integer, fname: string, deptid: integer)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class. Level(lvl) is a two character code with 4 different values (example: Junior: JR etc)

Write the following queries in SQL. No duplicates should be printed in any of the answers.

i. Find the names of all Juniors (level = JR) who are enrolled in a class taught by

ii. Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

iii. Find the names of all students who are enrolled in two classes that meet at the same time.

iv. Find the names of faculty members who teach in every room in which some class is taught.

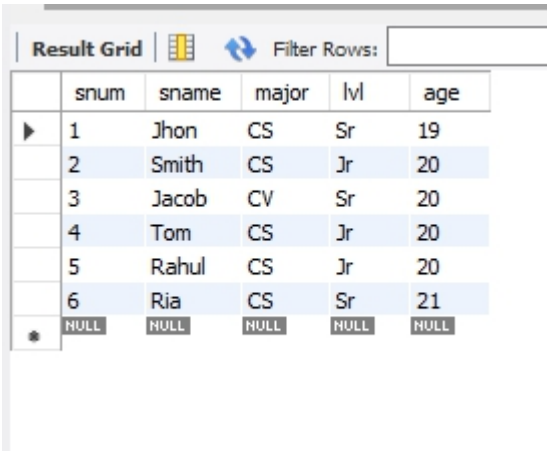
v. Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

vi. Find the names of students who are not enrolled in any class.

vii. For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

```
create database studentfaculty;
use studentfaculty;

create table student(
snum int not null,
sname varchar(20) not null,
major varchar(2) not null,
lvl varchar(2) not null,
age int not null,
primary key (snum)
);

create table faculty(
fid int not null,
fname varchar(20) not null,
deptid int not null,
primary key(fid)
);
```

```
create table class(
cname varchar(20) not null,
meetsat datetime not null,
room varchar(4) not null,
fid int not null,
primary key (cname),
foreign key(fid)references faculty(fid)
);

create table enrolled(
snum int not null,
cname varchar(20) not null,
primary key(snum,cname),
foreign key(snum)references student(snum),
foreign key(cname) references class(cname )
);

insert into student
 values (1,"Jhon","CS","Sr",19) ,(2,"Smith","CS","Jr",20),
(3,"Jacob","CV","Sr",20),(4,"Tom","CS","Jr",20),
(5,"Rahul","CS","Jr",20),(6,"Ria","CS","Sr",21);
```



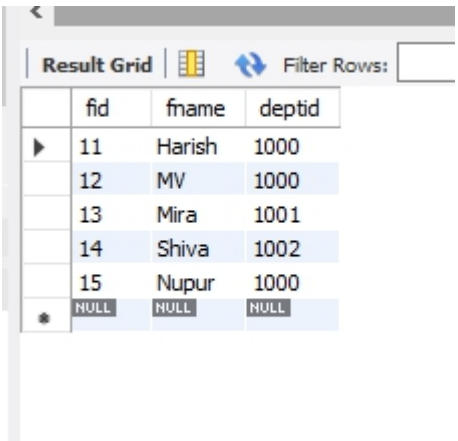
snum	sname	major	lvl	age
1	Jhon	CS	Sr	19
2	Smith	CS	Jr	20
3	Jacob	CV	Sr	20
4	Tom	CS	Jr	20
5	Rahul	CS	Jr	20
6	Ria	CS	Sr	21
NULL	NULL	NULL	NULL	NULL

insert into faculty

values(11,"Harish",1000),(12,"MV",1000),

(13,"Mira",1001),(14,"Shiva",1002),

(15,"Nupur",1000);



fid	fname	deptid
11	Harish	1000
12	MV	1000
13	Mira	1001
14	Shiva	1002
15	Nupur	1000
NULL	NULL	NULL

insert into class

values("class 1","2015-11-12 10:15:16","R1",14),

("class 10","2015-11-12 10:15:16","R128",14),

("class 2","2015-11-12 10:15:20","R2",12),

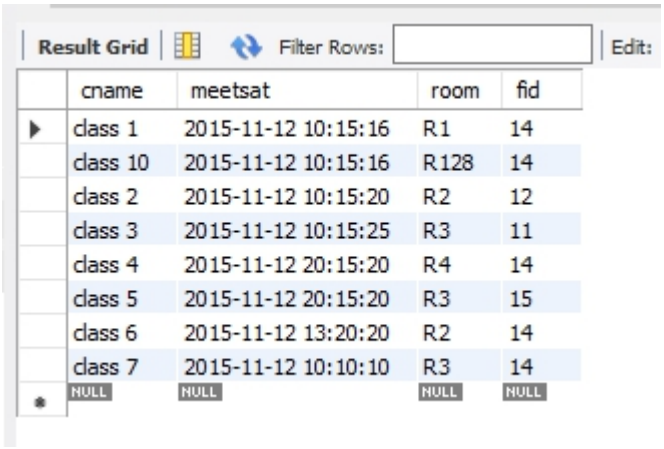
("class 3","2015-11-12 10:15:25","R3",11),

("class 4","2015-11-12 20:15:20","R4",14),

("class 5","2015-11-12 20:15:20","R3",15),

("class 6","2015-11-12 13:20:20","R2",14),

("class 7","2015-11-12 10:10:10","R3",14);



Result Grid | Filter Rows: | Edit:

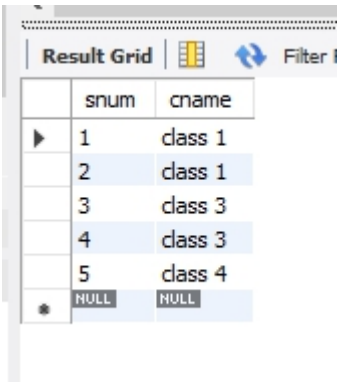
cname	meetsat	room	fid
class 1	2015-11-12 10:15:16	R1	14
class 10	2015-11-12 10:15:16	R128	14
class 2	2015-11-12 10:15:20	R2	12
class 3	2015-11-12 10:15:25	R3	11
class 4	2015-11-12 20:15:20	R4	14
class 5	2015-11-12 20:15:20	R3	15
class 6	2015-11-12 13:20:20	R2	14
class 7	2015-11-12 10:10:10	R3	14
NULL	NULL	NULL	NULL

insert into enrolled

values(1,"class 1"),(2,"class 1"),

(3,"class 3"),(4,"class 3"),

(5,"class 4");



Result Grid | Filter

snum	cname
1	class 1
2	class 1
3	class 3
4	class 3
5	class 4
NULL	NULL

------ Find the names of all Juniors (level = JR) who are enrolled in a class taught by

select s.sname from student s,enrolled e,class c

where s.snum=e.snum and c.cname = e.cname and c.fid =(select fid from faculty

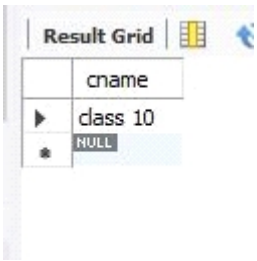
where fname ="Harish")and s.lvl="Jr";

------ Find the names of all classes that either meet in room R128 or have five or more Students enrolled.

select c.cname from class c

where c.room = "R128"

or c.cname in(select e.cname from enrolled e group by e.cname having count(*)>=5);



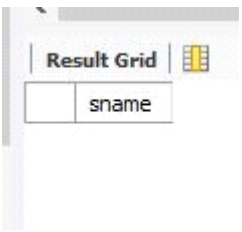
------ Find the names of all students who are enrolled in two classes that meet at the same time.

select distinct s.sname from student s

where s.snum in(select e1.snum from enrolled e1,enrolled e2,class c1,class c2

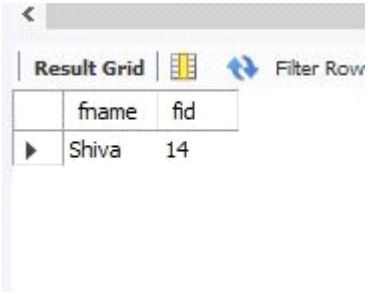
where e1.snum=e2.snum and e1.cname<>e2.cname and e1.cname = c1.cname

and e2.cname=c2.cname and c1.meetsat=c2.meetsat);



------ Find the names of faculty members who teach in every room in which some class is taught.

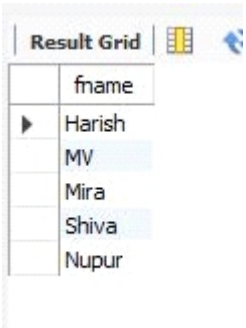
```
select f.fname,c.fid from faculty f,class c
 where f.fid = c.fid
 group by c.fid
 having count(c.fid)=(select count(distinct room) from class);
```



	fname	fid
▶	Shiva	14

------ Find the names of faculty members for whom the combined enrollment of the courses that they teach is less than five.

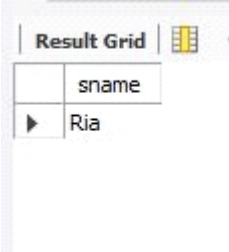
```
select distinct fname from faculty f
 where 5>(select count(e.snum)from enrolled e,class c
 where c.cname = e.cname and c.fid = f.fid);
```



	fname
▶	Harish
	MV
	Mira
	Shiva
	Nupur

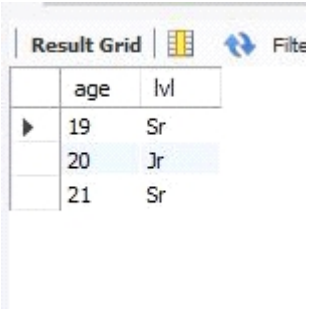
------ Find the names of students who are not enrolled in any class.

```
select s.sname from student s
 where snum not in(select snum from enrolled);
```



------ For each age value that appears in Students, find the level value that appears most often. For example, if there are more FR level students aged 18 than SR, JR, or SO students aged 18, you should print the pair (18, FR).

```
select s.age,s.lvl from student s
group by s.age having s.lvl in(select s1.lvl from student s1
where s1.age = s.age group by s1.age having count(*)>=all(select s2.lvl from student s2
where s2.age = s1.age group by s2.age));
```



PROGRAM 5

AIRLINE FLIGHT DATABASE

Consider the following database that keeps track of airline flight information:

FLIGHTS(flno: integer, from: string, to: string, distance: integer, departs: time, arrives: time, price: integer)

AIRCRAFT(aid: integer, aname: string, cruisingrange: integer)

CERTIFIED(eid: integer, aid: integer)

EMPLOYEES(eid: integer, ename: string, salary: integer)

Note that the Employees relation describes pilots and other kinds of employees as well; Every pilot is certified for some aircraft, and only pilots are certified to fly.

Write each of the following queries in SQL.

i. Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

ii. For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.

iii. Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

iv. For all aircraft with cruisingrange over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.

v. Find the names of pilots certified for some Boeing aircraft.

vi. Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

vii. A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.

```
create database airlines;

use airlines;


create table flights(

flno int not null,

from_loc varchar(20) not null,
```

```
to_loc varchar(20) not null,
distance int not null,
departs time not null,
arrives time not null,
price int not null,
primary key(flno)
);

create table aircraft(
aid int not null,
aname varchar(20) not null,
cruisingrange int not null,
primary key(aid)
);

create table employees(
eid int not null,
ename varchar(20) not null,
salary int not null,
primary key(eid)
);

create table certified(
eid int not null,
aid int not null,
```

primary key(eid,aid),

foreign key(eid)references employees(eid),

foreign key(aid)references aircraft(aid)

);

insert into flights

values(101,"Bangalore","Delhi",2500,"07:15:31","12:15:31",5000),

(102,"Bangalore","Lucknow",3000,"07:15:31","11:15:31",6000),

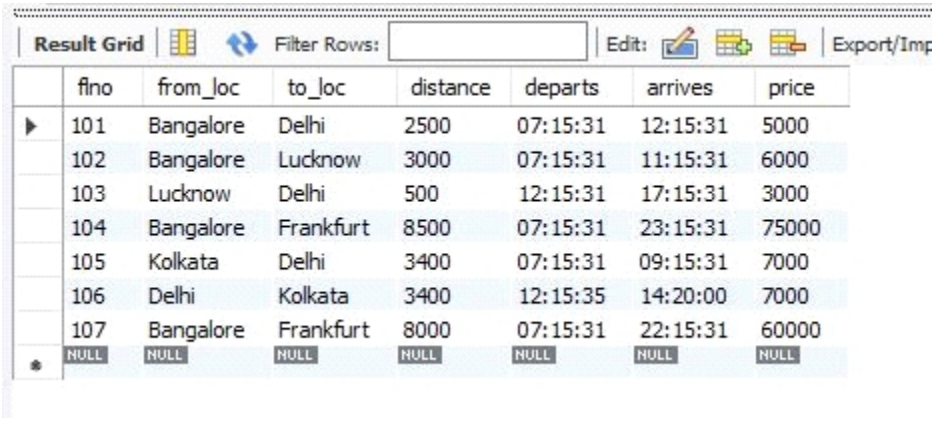
(103,"Lucknow","Delhi",500,"12:15:31","17:15:31",3000),

(107,"Bangalore","Frankfurt",8000,"07:15:31","22:15:31",60000),

(104,"Bangalore","Frankfurt",8500,"07:15:31","23:15:31",75000),

(105,"Kolkata","Delhi",3400,"07:15:31","09:15:31",7000),

(106,"Delhi","Kolkata",3400,"12:15:35","14:20:00",7000);


Result Grid | Filter Rows: | Edit: | Export/Imp

flno	from_loc	to_loc	distance	departs	arrives	price
101	Bangalore	Delhi	2500	07:15:31	12:15:31	5000
102	Bangalore	Lucknow	3000	07:15:31	11:15:31	6000
103	Lucknow	Delhi	500	12:15:31	17:15:31	3000
104	Bangalore	Frankfurt	8500	07:15:31	23:15:31	75000
105	Kolkata	Delhi	3400	07:15:31	09:15:31	7000
106	Delhi	Kolkata	3400	12:15:35	14:20:00	7000
107	Bangalore	Frankfurt	8000	07:15:31	22:15:31	60000
NULL	NULL	NULL	NULL	NULL	NULL	NULL

insert into aircraft

values (101,"747",3000),(102,"Boeing",900),

(103,"647",800),(104,"Dreamliner",10000),

(105,"Boeing",3500),(106,"707",1500),

(107,"Dream",12000);

Result Grid | Filter Rows:

aid	aname	cruisingrange
101	747	3000
102	Boeing	900
103	647	800
104	Dreamliner	10000
105	Boeing	3500
106	707	1500
107	Dream	12000
NULL	NULL	NULL

insert into employees

values(701,'A',50000),(702,'B',100000),

(703,'C',150000),(704,'D',90000),

(705,'E',40000),(706,'F',60000),

(707,'G',90000);

Result Grid | Filter Rows:

eid	ename	salary
701	A	50000
702	B	100000
703	C	150000
704	D	90000
705	E	40000
706	F	60000
707	G	90000
NULL	NULL	NULL

insert into certified

values(701,101),(701,102),

(701,106),(701,105),

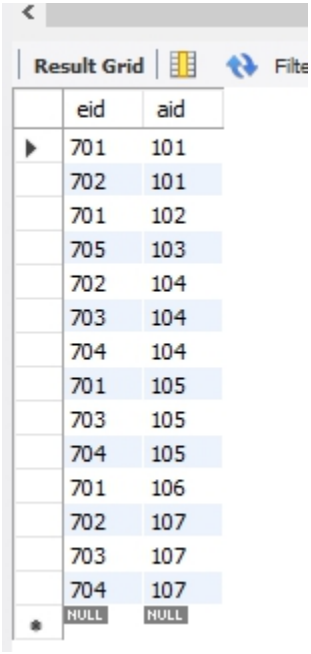
(702,104),(703,104),

(704,104),(702,107),

(703,107),(704,107),

(702,101),(703,105),

(704,105),(705,103);

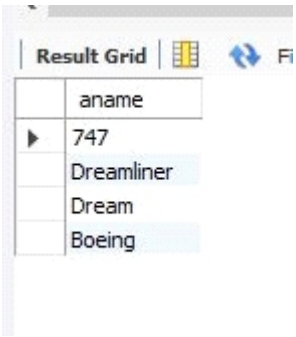


Result Grid

eid	aid
701	101
702	101
701	102
705	103
702	104
703	104
704	104
701	105
703	105
704	105
701	106
702	107
703	107
704	107
NULL	NULL

----- Find the names of aircraft such that all pilots certified to operate them have salaries more than Rs.80,000.

```
select distinct a.aname from aircraft a,employees e,certified c

where  a.aid =  c.aid and e.eid = c.eid and e.salary>80000;
```

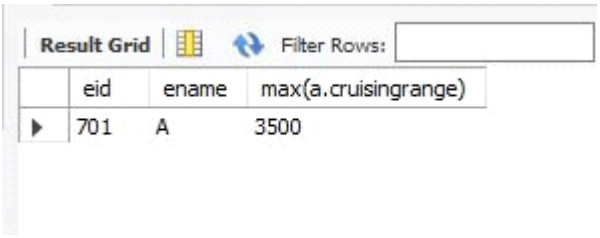


Result Grid

aname
747
Dreamliner
Dream
Boeing

----- For each pilot who is certified for more than three aircrafts, find the eid and the maximum cruisingrange of the aircraft for which she or he is certified.

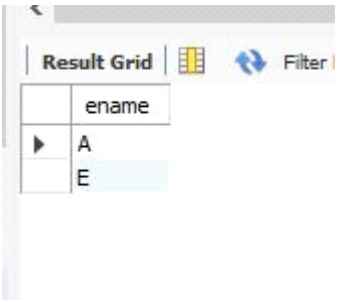
```
select e.eid ,e.ename,max(a.cruisingrange)from employees e,certified c,aircraft a

where e.eid = c.eid and a.aid = c.aid group by e.ename having count(c.aid)>3;
```



eid	ename	max(a.cruisingrange)
701	A	3500

----- Find the names of pilots whose salary is less than the price of the cheapest route from Bengaluru to Frankfurt.

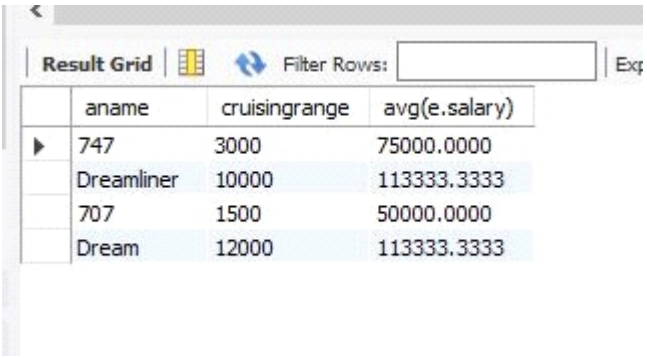
```
select e.ename from employees e

where salary<(select min(price) from flights

where from_loc = "Bangalore" and to_loc = "Frankfurt");
```



ename
A
E

----- For all aircraft with cruisingrange over 1000 Kms, find the name of the aircraft and the average salary of all pilots certified for this aircraft.

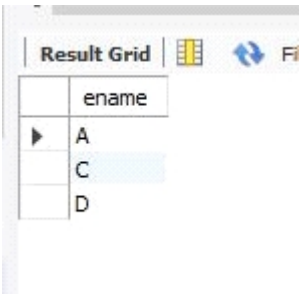
```
select a.aname,a.cruisingrange,avg(e.salary)

from aircraft a,employees e,certified c

where c.eid = e.eid and c.aid = a.aid

group by a.aname having a.cruisingrange>1000;
```



aname	cruisingrange	avg(e.salary)
747	3000	75000.0000
Dreamliner	10000	113333.3333
707	1500	50000.0000
Dream	12000	113333.3333

----- Find the names of pilots certified for some Boeing aircraft.

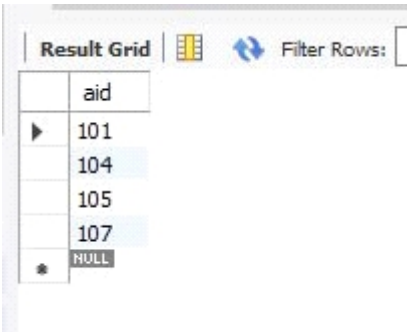
```
select distinct e.ename from employees e,certified c,aircraft a
where e.eid = c.eid and a.aid = c.aid and aname like "Boeing";
```



ename
A
C
D

----- Find the aids of all aircraft that can be used on routes from Bengaluru to New Delhi.

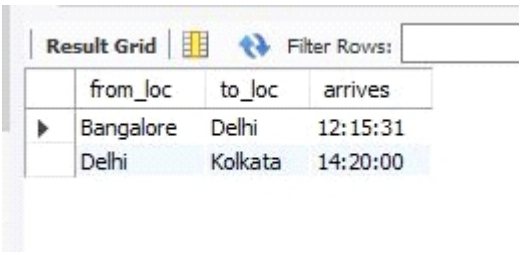
```
select a.aid from aircraft a
where a.cruisingrange>=(select distance from flights
where from_loc="Bangalore" and to_loc="Delhi");
```



aid
101
104
105
107
NULL

----- A customer wants to travel from Madison to New York with no more than two changes of flight. List the choice of departure times from Madison if the customer wants to arrive in New York by 6 p.m.

```
select f.from_loc,f.to_loc,f.arrives from flights f

where (f.from_loc ="Bangalore" and f.to_loc =(select from_loc from flights

where to_loc = "Kolkata")) or f.to_loc="Kolkata";
```



Result Grid | Filter Rows:

from_loc	to_loc	arrives
Bangalore	Delhi	12:15:31
Delhi	Kolkata	14:20:00